Pergamon

Computers Math. Applic. Vol. 32, No. 4, pp. 105–107, 1996

0898-1221/96 $15.00 + 0.00

S0898-1221(96)00128-9

NuMachine and NuAlgebra

K. K. NAMBIAR
School of Computer and Systems Sciences, Jawaharlal Nehru University
New Delhi 110067, India

(Received July 1995; accepted February 1996)

Abstract—A NuMachine capable of computing any recursive function is defined as a labeled graph with certain restrictions. A NuAlgebra that goes with the machine is also discussed with examples.

Keywords—NuMachine, NuAlgebra, Recursive functions.

1. INTRODUCTION

A directed graph, here called *NuMachine* (also written as *numachine*), can be used as a model of computation if the graph satisfies the properties given below.

1. All edges have one of the three labels a, b, or e. Two different edges can have the same label.
2. All nodes have labels $1, 1', 1'', \ldots, 2, 2', 2'', \ldots, 3, \ldots, \ldots, 0$, the last label always being zero. No two different nodes can have the same label. Node 1 is called the initial node and node 0 is called the final node.
3. The outdegree of the node 0 is zero. The outdegree of every other node is either one or two. If it is one, the outgoing edge is labeled as a, and if it is two, the outgoing edges are labeled as b and e.
4. A set of nodes is designated as input nodes and a single node is designated as the output node.

The basis for the definition of this graph is the *abacus* machine described in [1], where it is shown that abacus machines are equivalent to Turing machines. Since numachine is nothing but a formalized version of the abacus machine, defined in terms of a graph, it follows that numachines are also equivalent to Turing Machines and can compute any recursive function.

Figures 1 and 2 are examples of numachines which can perform addition and multiplication, respectively. For the adding machine, the input nodes are 1 and 2 and the output node is 2, and for the multiplying machine, the input nodes are 1 and 2 and the output node is 3.

2. NUMACHINE

The working of the machine can be easily understood if we imagine each node as a register having a string of a's stored in it. The machine always starts from node 1 and moves from node to node depending on the outgoing edges of the node. If the outgoing edge has label a, then the machine increments the length of the string at that node by one and goes along the edge a to the next node. If the outgoing edges have the labels b and e, then the length of the string is decremented by one and goes along the edge b to the next node. If the length of the string is already zero and hence cannot be decremented, the machine simply moves along edge e to the next node.

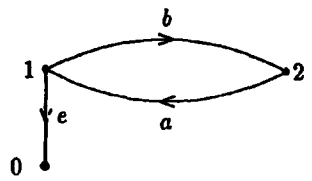

Figure 1.

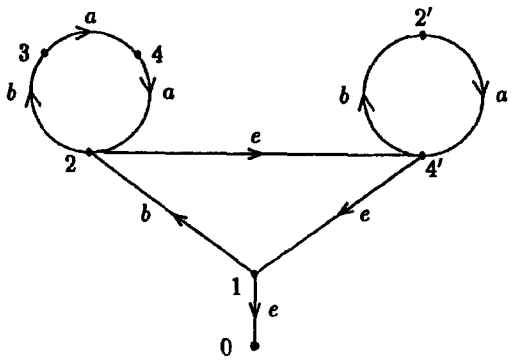

Figure 2.

When a set of nodes have the same number for a label, ignoring primes, it means that the strings in those nodes can be incremented or decremented only simultaneously. When the string in one of the nodes in the set is incremented or decremented, all the other nodes in the set do so automatically. The working of the multiplication machine in Example 2, given later, illustrates this. If the machine ever reaches the node 0, then it just stays put there and the computation is considered to be over.

3. NUALGEBRA

We use the adjacency matrix [2] of the numachine in our calculations in the *NuAlgebra* (also written as *nualgebra*). The adjacency matrix operates on a *state vector* of the machine and produces a new state vector, and this process continues until the operation is no longer possible. The multiplication of matrices in nualgebra is totally different from the usual one in that at every stage the move to the next state vector is decided by just one row, and that also the row affects only the corresponding row of the state vector. The position of the nonzero elements in the row decide which row of the adjacency matrix is to be used next. The multiplication always starts off with the first row of the matrix. These facts can easily be inferred from the working of the numachine.

The two examples below illustrate the transitions of the machines in Figures 1 and 2 for addition and multiplication. We will use the notation $[k]$ for the length of the string in node k.

EXAMPLE 1. In this example, the numachine calculates $2+2$ as 4. The initial state vector shows that $[1] = 2$ and $[2] = 2$. The final state vector shows $[2] = 4$.

$$\begin{array}{c} \\ 1 \\ 2 \\ 0 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} 1 & 2 & 0 \end{array} \\ \begin{pmatrix} & b & e \\ a & & \\ & & \end{pmatrix} \end{array} \begin{pmatrix} a^2 \\ a^2 \\ e \end{pmatrix} = \begin{array}{ccc} a & a & e \\ a^2 & a^3 & a^3 \\ e & e & e \end{array} \begin{pmatrix} e \\ a^4 \\ e \end{pmatrix}.$$

EXAMPLE 2. Here the machine calculates 2×2 as 4. The initial state vector shows that $[1] = 2$ and $[2] = 2$. The final state vector shows $[3] = 4$.

$$\begin{array}{c} \\ 1 \\ 2 \\ 2' \\ 3 \\ 4 \\ 4' \\ 0 \end{array}\!\!\begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 2' & 3 & 4 & 4' & 0 \end{array} \\ \begin{pmatrix} & b & & & & & e \\ & & & b & & e & \\ & & & & & a & \\ & & & a & & & \\ & a & & & & & \\ e & & b & & & & \\ & & & & & & \end{pmatrix} \end{array} \begin{pmatrix} a^2 \\ a^2 \\ a^2 \\ e \\ e \\ e \\ e \end{pmatrix} = \begin{array}{ccccccccccccc} a & a & a & a & a & a & a & a & a & a & a & e & \cdots \\ a^2 & a & a & a & e & e & e & e & a & a & a^2 & a^2 & \cdots \\ a^2 & a & a & a & e & e & e & e & a & a & a^2 & a^2 & \cdots \\ e & e & a & a & a & a^2 & a^2 & a^2 & a^2 & a^2 & a^2 & a^2 & \cdots \\ e & e & e & a & a & a & a^2 & a & a & e & e & e & \cdots \\ e & e & e & a & a & a & a^2 & a & a & e & e & e & \cdots \\ e & e & e & e & e & e & e & e & e & e & e & e & \cdots \end{array} \begin{pmatrix} e \\ a^2 \\ a^2 \\ a^4 \\ e \\ e \\ e \end{pmatrix}.$$

4. CONCLUSION

The attempt here has been to show that computation of recursive functions can also be viewed as matrix multiplication apart from pebble manipulation and string processing.

REFERENCES

1. G. Boolos and R. Jeffrey, *Computability and Logic*, Cambridge University Press, New York, (1974).
2. N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice Hall of India, New Delhi, (1984).